Valentina Stoma

DATA5020 Collect/Store/Retrieve Data Sec 01

Fall 2019

Professor: Sara Arunagiri

Purpose of the Project

For the purpose of this project, I chose to focus on examining the National Parks in the US for convenient general purpose use. National parks in the US are some of the most visited natural attractions not only in the US but for the rest the world as well (Western U.S. National Parks Popular with Foreign Tourists). Thousands of people from all over the world visit the US every year to hike up the famous mountains, see the Grand Canyon, and visit the oldest National Park in the US – Yellowstone National Park, established 18972 (Wee).The popularity of many locations all over the country bring tourism to the states that would not normally attract the same number of tourists. Based on this interest in the US National Parks, the project is focused on developing a comprehensive database with the list of National Parks in the US visited by Americans and international tourists. Some of the information that could be useful for trip planning purposes includes the list of National parks, state location, date it was established, and the acres size of the park. Additionally, I am interested in the minimal elevation level, as well as the vertical relief (the change in elevation that occurs from the ground level to the peak). Apart from the statistical information regarding the elevation levels, I am interested in the Mountain range and the Peak that is the official highest elevation for that mountain range.

In addition to described general information for every national park, the report covers a number of trails – assessing the difficulty of them and providing a succinct classification – for some of the oldest and most visited parks. For the purpose of this information to be useful for the tourists outside of the US, I cover three of some of the most internationally visited National Parks – Yellowstone, Yosemite, and Olympic (Western U.S. National Parks Popular with Foreign Tourists). For Yosemite, the project includes an accident report in order to provide a template for future reports. Currently, this is not a normal practice to keep track of all accidents for every national park, therefore, this modification can be applied for the future database. At the moment, news can be accessed for every national park on the curated list,

however, I am not including this data in the following database as it is not updated regularly for all the parks on the list (News Releases (U.S. National Park Service)).

Level above general visiting plans, this project is interested in obtaining, keeping, and tracking information about yearly recreational visitors to National Parks, natural protected areas, and natural historic sites in the US. That dataset is obtained from the National Park Service Visitor User Statistics (National Parks Service) - a great resource for reported data for different types of visitations that are recorded at the parks. For this project, the databased contains only recreational visits. Data is obtained for the ten year period from 2009 until 2018, inclusive. This dataset covers many more natural and historic sites, compared to the easily publicly available Natural Park list.

This project connects two previously disconnected parts of data from National Park Services and publicly available sources. This database allows to perform basic analysis of travel trends, as well as general planning by general population.

Collection of Data and Implementation

Problems that can be addressed with this data include assessing national park trip planning based on the location, popularity (visitor number), and trail information of the national parks in the US. Additionally, trends in national park visitations can be estimated. At the moment, changes in visitor number of the popular national parks over the last 10 years can be demonstrated. Additionally, since there is a lot of open-source information about the dates that the parks were established or visitor number, some sources can have discrepancies. Utilizing alternative sources and verified National Park Service information, the database can provide a demonstration of inconsistencies and misreported data. Moreover, utilizing obtained data, the report visualizes the peak height information (Appendix, Image 3). Additionally, trends

in visitations to different National Parks can be visualized – attached are examples of individual and combined visualization that can be performed on the obtained data. (Appendix, Image 14,15). This data can be well visualized spacially, example is attached in Appendix A, Image 16.

Methods:

Data for this project was obtained from a number of different open sources. First, as one of the goals of this database is to assists in National Park trip planning, I use Wikipedia as a popular first source for general public (List of National Parks of the United States.). The list is succinct and contains only National Park data, in comparison to other sources that combine historic sites, reservoirs, and other types of locations that fall under National Park Service. I combined multiple ways of obtaining publicly available data, depending on which was the most efficient for the layout of the website. For Wikipedia list, I extracted data using Import.io (Web Data Integration – Import.io - Data Extraction, Web Data Harvesting, Data Preparation, Data Integration). Image of who the not cleaned up data looks uploaded into R is attached in the Appendix (Image 1). Upon cleaning up the data, I obtain a dataframe (clean_parks) that can be uploaded to the database (Image 2). Additionally, I create a copy of the clean_parks dataframe, which contains a column with abbreviations of State names, which can be used for graph generation.

Another dataset, which contains 10 most visited parks was extracted from publicly available data on National Park Service website (Visitation Numbers (U.S. National Park Service)). This website data was used to compare to the National Park Service Full report for all historic, reservoir, park, and other location visitations over the last 10 years. This dataset was obtained utilizing R package **rvest** and cleaned up using **dplyr** and **stringr**. Cleaned up dataset sample is attached in Appendix (Image 4)

Next dataset is included in this project with the goal to demonstrate the absence of comprehensive accessible accident report for all the National Parks in the US. This dataset is a report of accidents in

Yosemite National Park from 2012 till 2018 (Search and Rescue: Lessons from the Field). Image of the dataset is attached in Appendix (Image 5) . This dataset does not satisfy the requirement to be included into the database because of absence of unique identifier, since the only National park included in this set is Yosemite. This data allows us to plot a histogram of the accidents, which is attached in the Appendix (Image 6)

For the purpose of trip planning, the database includes a dataset with trail information for three out of the most visited National Parks in the US by tourists (Western U.S. National Parks Popular with Foreign Tourists). This dataset contains combined information from a number of publicly available sources (12 Day Hikes in Yellowstone) (Yosemite Valley Day Hikes) (Olympic National Park Hiking Trails). Yellowstone and Yosemite trail information (name and difficulty level) was extracted using rvset package. Olympic park trail information (names and difficulty level) was extracted using Import.io (Web Data Integration – Import.io - Data Extraction, Web Data Harvesting, Data Preparation, Data Integration). Image of the final dataset, containing there National Park subset of trails is attached in Appendix (Image 7).

Elevation dataset was extracted using Import.io (Web Data Integration – Import.io - Data Extraction, Web Data Harvesting, Data Preparation, Data Integration) (List of National Parks of the United States.) Within the elevation dataset, final elevation at the Peak was calculated within dataframe, as the initial data input lacked the final elevation value. Image of the initial and the cleaned up dataframes are attached in the Appendix (Image 8,9).

Additionally, publicly available dataset with 10 oldest parks in the US was retrieved with the goal of comparing the information available on Wikipedia with other publicly available referenced sources (Wee). This dataset was extracted using **rvest** R package.

Finally, a comprehensive report of every National Park tracked location and visitor number was obtained as an xlsx file from The Integrated Resource Management Applications (IRMA) National Park data (Stats Report Viewer). The file could not be extracted in cvs format, which would be the preferred option , but National Park Service Reports that: "*This report will not correctly export to CSV. Please export to Excel, Word, or PDF if you wish to save this report".*

Upon reading the file into R, the dataset had to be significantly cleaned up. R package **stringr** was used to ensure that the National Park names extracted from the different sources are equal. Cleaned up version of the report in attached in the Appendix (Image 11). The data was cleaned up to move different year values from individual columns into one "Year" column with corresponding visitor values in appropriate rows. That was achieved with command **gather** from **tidyr**.

Issues and Resolutions:

One of the issues encountered upon obtaining data from multiple data sources on the internet and uploading them to the SQL database, was that there was internal formatting in different datasets, which prohibited different tables from Joining based on equal values. In order to correct for this, careful **stringr** editting was performed. Additionally, in different datasets different number of leading and lagging spaces was found, which also prohibited SQL operations on multiple tables. These inconsistencies in spaces were trimmed with **str_trim** from **stringr**.

Another issue was encountered with uploading dataframes which had date format into SQL database. If the column in the database was formatted as Date, upon reading the df in the SQL db, the values were formatted in a way that made them untrue. In order to avoid that, the columns formatted as Date needed to be formatted into character. However, when creating a table in SQL, the variable should be specified

as DATE. With these modifications, the table inside the database is formatted correctly – example is attached in Appendix 1 (Image 12).

Since data was obtained from open sources with initial formatting, there were other formatting issue encountered when changing the format of different columns. For example, one such issue was in the process of converting minimum elevation for National parks into an integer in order to calculate the elevation of the Peak. For Death Valley, minimal elevation is below sea level, which is represented by a negative sign. The format of the negative sign was different in the source data and upon transformation, the final Peak height was calculated as NA because the negative value could not be converted and NA was propagated. That resulted in an initial incorrect plot generation (Appendix A, Image 13). This was corrected and plot with no missing values was generated (Appendix A – Image 3).

There was an issue encountered with pdf generation from the R document. Upon having a complete document, I received an error that halted the pdf generation with the message that Unicode character − (U+2212) was interrupting the pdf knitting. This indicated that this line:

```
#mutate(Min_elevation_feet = str_replace(Min_elevation_feet, "−", "-")) %>%
```

In my code cleaning upon elev_all_parks dataframe was interrupting the process. Based on this, in the pdf of the project, that line is absent. The Rmd document contains the line.

Insights and future work:

Over the course of the work on this project, I have collected significant amount of information which can be useful for trip planning, as well as visitation trend analysis and projections. I have also uncovered a missing piece of data, which can be collected and obtained as a part of future work – many national parks have no record of accidents and safety concerns, which can be easily traced back. Although there are

current news updates for most National Parks in the US, there is no non-pdf record of accidents and park police reports. Since National Parks often have internal sheriffs, state police does not always have records for the National Park accidents either.

Moreover, future work can include supplementing this database with weather information (temperature change over the years) to estimate temperature changes in different areas of the country in the national parks, and how that temperature difference has affected the visitation number. I have found reports on the visitors number increasing with the increased temperatures in certain National Parks (100 Years of Warming in the National Parks), however, this database update is a part of future work.

Appendix A – Images

	Name	State	Date_established	Area
1	Acadia	Maine	February 26, 1919	49,075.26 acres (198.6 km
2	American Samoa	American Samoa	October 31, 1988	8,256.67 acres (33.4 km
3	Arches	Utah	November 12, 1971	76,678.98 acres (310.3 km
4	Badlands	South Dakota	November 10, 1978	242,755.94 acres (982.4 km
5	Big Bend	Texas	June 12, 1944	801,163.21 acres (3,242.2 km
6	Biscayne	Florida	June 28, 1980	172,971.11 acres (700.0 km
7	Black Canyon of the Gunnison	Colorado	October 21, 1999	30,779.83 acres (124.6 km
8	Bryce Canyon	Utah	February 25, 1928	35,835.08 acres (145.0 km
9	Canyonlands	Utah	September 12, 1964	337,597.83 acres (1,366.2 km
10	Capitol Reef	Utah	December 18, 1971	241,904.50 acres (979.0 km
11	Carlsbad Caverns	New Mexico	May 14, 1930	46,766.45 acres (189.3 km
12	Channel Islands	California	March 5, 1980	249,561.00 acres (1,009.9 km
13	Congaree	South Carolina	November 10, 2003	26,639.99 acres (107.8 km
14	Crater Lake	Oregon	May 22, 1902	183,224.05 acres (741.5 km
15	Cuyahoga Valley	Ohio	October 11, 2000	32,571.88 acres (131.8 km
16	Death Valley	California	October 31, 1994	3,372,981.42 acres (13,650.0 km

Showing 1 to 17 of 61 entries, 4 total columns

Image 1 – Figure demonstrates initial view of the “parks” dataframe before the clean up – obtained from (List of National Parks of the United States.)

	Name	State	Date_established	Area_acres
1	Acadia	Maine	1919-02-26	4907526
2	American Samoa	American Samoa	1988-10-31	825667
3	Arches	Utah	1971-11-12	7667898
4	Badlands	South Dakota	1978-11-10	24275594
5	Big Bend	Texas	1944-06-12	80116321
6	Biscayne	Florida	1980-06-28	17297111
7	Black Canyon of the Gunnison	Colorado	1999-10-21	3077983
8	Bryce Canyon	Utah	1928-02-25	3583508
9	Canyonlands	Utah	1964-09-12	33759783
10	Capitol Reef	Utah	1971-12-18	24190450
11	Carlsbad Caverns	New Mexico	1930-05-14	4676645
12	Channel Islands	California	1980-03-05	24956100
13	Congaree	South Carolina	2003-11-10	2663999
14	Crater Lake	Oregon	1902-05-22	18322405
15	Cuyahoga Valley	Ohio	2000-10-11	3257188
16	Death Valley	California	1994-10-31	3372,98142
17	Denali	Alaska	1917-02-26	4740,91116
18	Dry Tortugas	Florida	1992-10-26	6470122
19	Everglades	Florida	1934-05-30	1508,93857
20	Gates of the Arctic	Alaska	1980-12-02	7523,89745
21	Gateway Arch	Missouri	2018-02-22	19283
22	Glacier	Montana	1910-05-11	1013,12599

Showing 1 to 23 of 61 entries, 4 total columns

Image 2 – Figure contains the view of the final “clean_parks” dataframe as called from R.

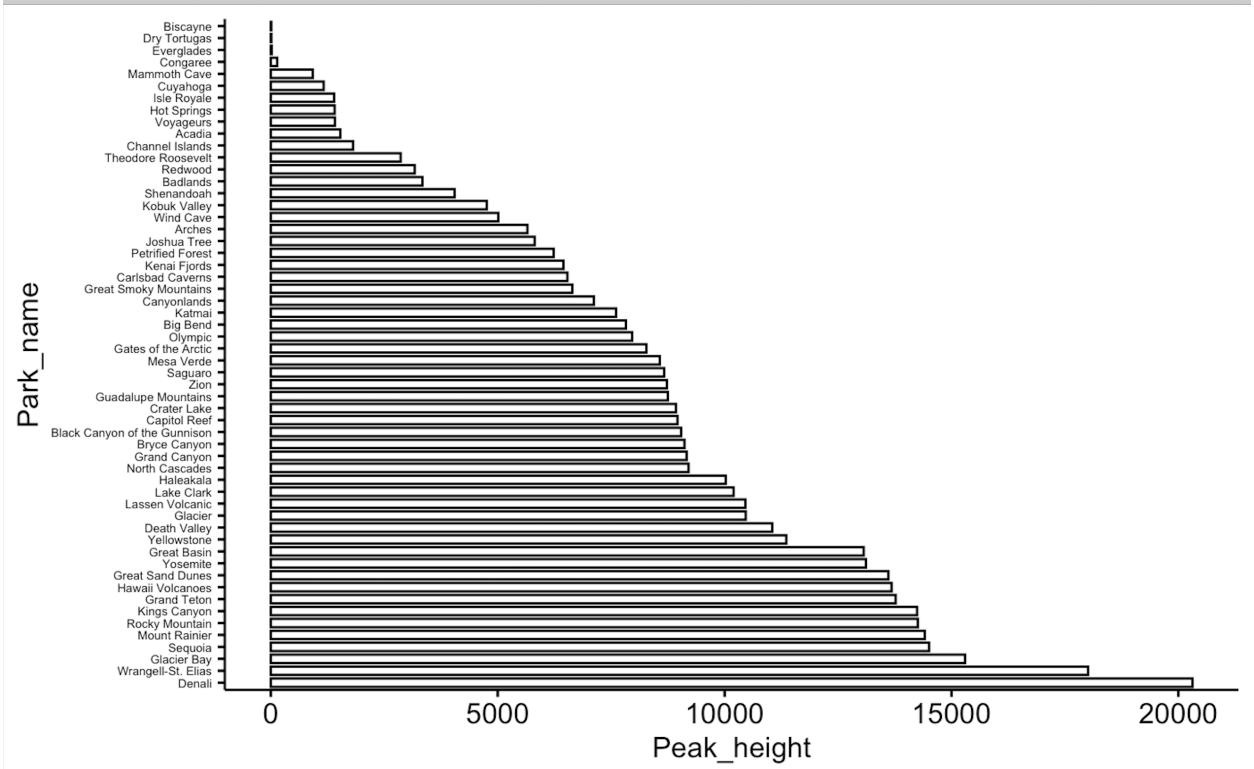


Image 3 – Figure demonstrates the peak height for the provided national parks, according this the extracted peak height information. (List of National Parks of the United States.)

	Park_name	visitor_number
1	Great Smoky Mountains	15223697
2	Grand Canyon	14690418
3	Rocky Mountain	11421200
4	Zion	9243305
5	Yellowstone	7804683
6	Yosemite	7578958
7	Acadia	7288623
8	Grand Teton	6380495
9	Olympic	6362439
10	Glacier	4719148

Image 4 – Dataframe which contains top 10 most visited National Parks in the US in 2018.

	yose_name	Yosemite_accident_dates	accident_description
1	Yosemite	2018-10-15	When the Merced RIver water level drops midsummer,...
2	Yosemite	2018-10-01	While descending from Half Dome, staying on the trai...
3	Yosemite	2018-09-05	On the afternoon of June 24, 2017, a hiker slipped on...
4	Yosemite	2018-07-17	Late in the morning of Friday, June 8, 2018, the Yose...
5	Yosemite	2018-04-29	On Tuesday, February 20, 2018, Alan Chow was repor...
6	Yosemite	2017-11-12	On the evening of Friday, August 12, 2016, two youn...
7	Yosemite	2017-07-06	On July 1, 2017 about an hour before sunset, a wilder...
8	Yosemite	2017-06-22	On May 16, a mother and daughter were hiking to Ins...
9	Yosemite	2017-05-15	On Friday, April 28, 2017, around 1 pm, the Big Trees...
10	Yosemite	2017-05-01	On the afternoon of April 29, a 39-year-old hiker abo...
11	Yosemite	2017-04-07	We normally have stories about people making choice...
12	Yosemite	2017-03-29	At 8 am on December 28, 2016, two trail runners—Ja...
13	Yosemite	2016-09-13	The following incident was provided to us from a visit...
14	Yosemite	2016-09-13	The following incident was provided to us from a visit...
15	Yosemite	2016-07-29	A brief dip in the Emerald Pool became a near-death ...
16	Yosemite	2016-07-17	The common denominator in many outdoor mishaps, ...
17	Yosemite	2016-06-12	The current Sierra snowpack melt is a welcome occurr...
18	Yosemite	2016-06-12	On Monday, June 6, 2016, Yosemite Search and Rescu...
19	Yosemite	2016-02-14	The Nose is considered to be the easiest full-length r...
20	Yosemite	2015-11-17	On Monday, November 2, 2015, winter arrived in Yos...
21	Yosemite	2015-10-08	On Monday, September 19, 2015, at approximately 5:...
22	Yosemite	2015-09-11	If you find yourself in a helicopter over the Yosemite I...

Showing 1 to 23 of 69 entries, 3 total columns

Image5 – Figure of the dataframe which contains example of future data to be recorded and obtained from all national parks. The dataframe contains the name of the National park, the date of the accident, and the accident description.

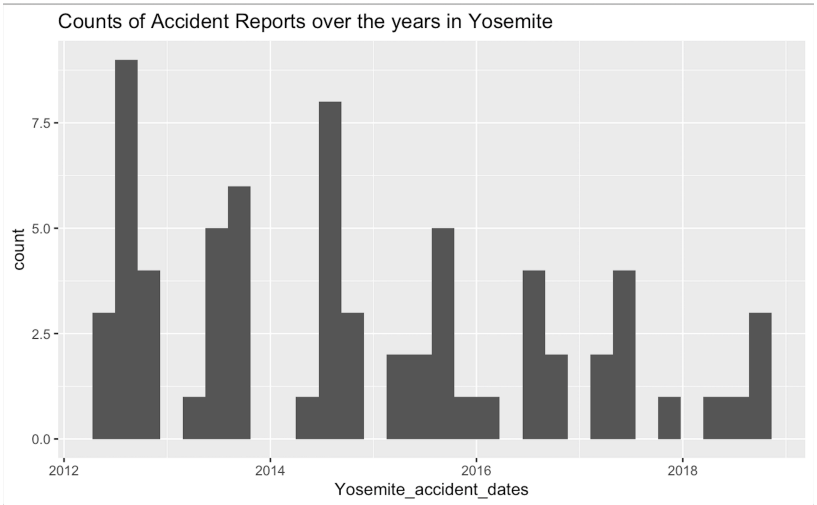


Image 6 - Counts of Accident Reports over the years in Yosemite

	park	trail_name	difficulty
1	Yellowstone	Storm point loop trail	Easy
2	Yellowstone	Trout lake	Easy
3	Yellowstone	Mystic falls	Easy
4	Yellowstone	Tower fall overlook	Easy
5	Yellowstone	Uncle tom's trail (from artist point)	Moderate
6	Yellowstone	Elephant back mountain trail	Moderate
7	Yellowstone	Beaver ponds loop	Moderate
8	Yellowstone	Lone star geyser	Easy
9	Yellowstone	South rim of the canyon	Moderate
10	Yellowstone	Bunsen peak trail	Strenuous
11	Yellowstone	Mount washburn trail	Strenuous
12	Yellowstone	Avalanche peak	Strenuous
13	Yosemite	Bridalveil Fall Trail	Easy
14	Yosemite	Lower Yosemite Fall Trail	Easy
15	Yosemite	Cook's Meadow Loop	Easy
16	Yosemite	Mirror Lake Loop	Moderate
17	Yosemite	Valley Loop Trail	Moderate
18	Yosemite	Vernal Fall and Nevada Fall Trails	Strenuous
19	Yosemite	Yosemite Falls Trail	Strenuous
20	Yosemite	Snow Creek Trail	Strenuous
21	Yosemite	Four Mile Trail	Strenuous
22	Yosemite	Half Dome Trail	Strenuous

Showing 1 to 23 of 31 entries, 3 total columns

Image 7 – Figure of data frame, which contains National Park name, trail name, and the description of the level of difficulty of that trail.

	Park_name	Peak_name	Mountain_range	Mim_elevation	Vertical.relief
1	Denali	Denali	Alaska Range	240 feet (73 m)	20,070 feet (6,120 m)
2	Wrangell–St. Elias	Mount Saint Elias	Saint Elias Mountains	0 feet (0 m)	18,008 feet (5,489 m)
3	Glacier Bay	Mount Fairweather	Saint Elias Mountains	0 feet (0 m)	15,300 feet (4,700 m)
4	Sequoia	Mount Whitney	Sierra Nevada	1,360 feet (410 m)	13,145 feet (4,007 m)
5	Mount Rainier	Mount Rainier	Cascade Range	1,610 feet (490 m)	12,801 feet (3,902 m)
6	Rocky Mountain	Longs Peak	Front Range	7,630 feet (2,330 m)	6,629 feet (2,021 m)
7	Kings Canyon	North Palisade	Sierra Nevada	3,480 feet (1,060 m)	10,762 feet (3,280 m)
8	Grand Teton	Grand Teton	Teton Range	6,310 feet (1,920 m)	7,460 feet (2,270 m)
9	Hawaii Volcanoes	Mauna Loa	Hawaiian Islands	0 feet (0 m)	13,679 feet (4,169 m)
10	Great Sand Dunes	Tijeras Peak	Sangre de Cristo Range	7,520 feet (2,290 m)	6,090 feet (1,860 m)
11	Yosemite	Mount Lyell	Sierra Nevada	2,105 feet (642 m)	11,009 feet (3,356 m)
12	Great Basin	Wheeler Peak	Snake Range	6,195 feet (1,888 m)	6,870 feet (2,090 m)
13	Yellowstone	Eagle Peak	Absaroka Range	5,282 feet (1,610 m)	6,076 feet (1,852 m)
14	Death Valley	Telescope Peak	Panamint Range	−279 feet (−85 m)[5]	11,328 feet (3,453 m)
15	Glacier	Mount Cleveland	Lewis Range	3,150 feet (960 m)	7,316 feet (2,230 m)
16	Lassen Volcanic	Lassen Peak	Cascade Range	5,275 feet (1,608 m)	5,182 feet (1,579 m)
17	Lake Clark	Redoubt Volcano	Aleutian Range	0 feet (0 m)	10,197 feet (3,108 m)
18	Haleakala	Haleakala	Hawaiian Islands	0 feet (0 m)	10,023 feet (3,055 m)
19	North Cascades	Goode Mountain	Cascade Range	605 feet (184 m)	8,601 feet (2,622 m)
20	Grand Canyon	North Rim	Kaibab Plateau	1,173 feet (358 m)	7,992 feet (2,436 m)
21	Bryce Canyon	Rainbow Point	Paunsaugunt Plateau	6,565 feet (2,001 m)	2,550 feet (780 m)
22	Black Canyon of the Gunnison	Poison Spring Hill	Grand Mesa	5,440 feet (1,660 m)	3,600 feet (1,100 m)

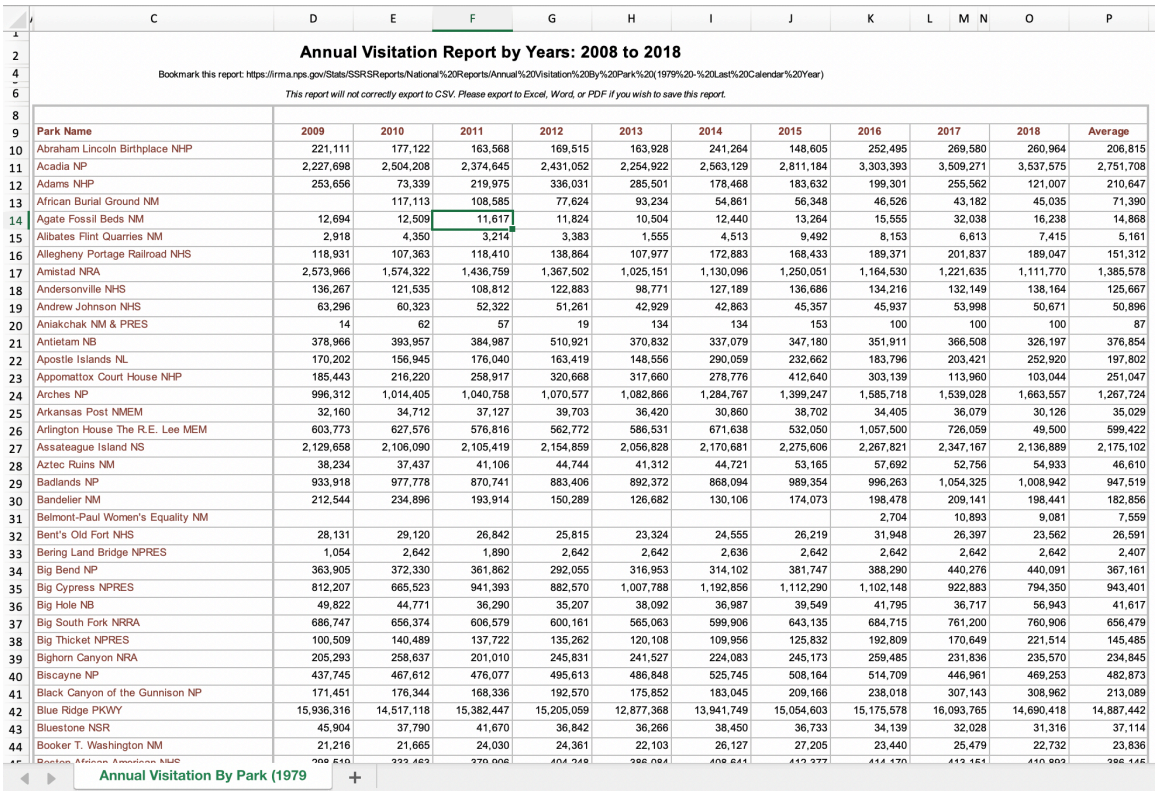
Showing 1 to 23 of 56 entries, 5 total columns

Image 8 - Figure contains initial view of the “elevation” dataframe before clean up .

	Park_name	Peak_name	Mountain_range	Min_elevation_feet	Vertical_relief_feet	Peak_height
1	Denali	Denali	Alaska Range	240	20070	20310
2	Wrangell–St. Elias	Mount Saint Elias	Saint Elias Mountains	0	18008	18008
3	Glacier Bay	Mount Fairweather	Saint Elias Mountains	0	15300	15300
4	Sequoia	Mount Whitney	Sierra Nevada	1360	13145	14505
5	Mount Rainier	Mount Rainier	Cascade Range	1610	12801	14411
6	Rocky Mountain	Longs Peak	Front Range	7630	6629	14259
7	Kings Canyon	North Palisade	Sierra Nevada	3480	10762	14242
8	Grand Teton	Grand Teton	Teton Range	6310	7460	13770
9	Hawaii Volcanoes	Mauna Loa	Hawaiian Islands	0	13679	13679
10	Great Sand Dunes	Tijeras Peak	Sangre de Cristo Range	7520	6090	13610
11	Yosemite	Mount Lyell	Sierra Nevada	2105	11009	13114
12	Great Basin	Wheeler Peak	Snake Range	6195	6870	13065
13	Yellowstone	Eagle Peak	Absaroka Range	5282	6076	11358
14	Death Valley	Telescope Peak	Panamint Range	-279	11328	11049
15	Glacier	Mount Cleveland	Lewis Range	3150	7316	10466
16	Lassen Volcanic	Lassen Peak	Cascade Range	5275	5182	10457
17	Lake Clark	Redoubt Volcano	Aleutian Range	0	10197	10197
18	Haleakala	Haleakala	Hawaiian Islands	0	10023	10023
19	North Cascades	Goode Mountain	Cascade Range	605	8601	9206
20	Grand Canyon	North Rim	Kaibab Plateau	1173	7992	9165
21	Bryce Canyon	Rainbow Point	Paunsaugunt Plateau	6565	2550	9115
22	Black Canyon of the Gunnison	Poison Spring Hill	Grand Mesa	5440	3600	9040

Showing 1 to 23 of 56 entries, 6 total columns

Image 9 – Figure contains initial view of the “elev_all_parks” dataframe after the clean up – final version.



Annual Visitation Report by Years: 2008 to 2018

Bookmark this report: https://irma.nps.gov/Stats/SSRSReports/National%20Reports/Annual%20Visitation%20By%20Park%20(1979%20-%20Last%20Calendar%20Year)

This report will not correctly export to CSV. Please export to Excel, Word, or PDF if you wish to save this report.

Park Name	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	Average
Abraham Lincoln Birthplace NHP	221,111	177,122	163,568	169,515	163,928	241,264	148,605	252,495	269,580	260,964	206,815
Acadia NP	2,227,698	2,504,208	2,374,645	2,431,052	2,254,922	2,563,129	2,811,184	3,303,393	3,509,271	3,537,575	2,751,708
Adams NHP	253,656	73,339	219,975	336,031	285,501	178,468	183,632	199,301	255,562	121,007	210,647
African Burial Ground NM		117,113	108,585	77,624	93,234	54,861	56,348	46,526	43,182	45,035	71,390
Agate Fossil Beds NM	12,694	12,509	11,617	11,824	10,504	12,440	13,264	15,555	32,038	16,238	14,868
Alibates Flint Quarries NM	2,918	4,350	3,214	3,383	1,555	4,513	9,492	8,153	6,613	7,415	5,161
Allegheny Portage Railroad NHS	118,931	107,363	118,410	138,864	107,977	172,883	168,433	189,371	201,837	189,047	151,312
Amistad NRA	2,573,966	1,574,322	1,436,759	1,367,502	1,025,151	1,130,096	1,250,051	1,164,530	1,221,635	1,111,770	1,385,578
Andersonville NHS	136,267	121,535	108,812	122,883	98,771	127,189	136,686	134,216	132,149	138,164	125,667
Andrew Johnson NHS	63,296	60,323	52,322	51,261	42,929	42,863	45,357	45,937	53,998	50,671	50,896
Aniakchak NM & PRES	14	62	57	19	134	134	153	100	100	100	87
Antietam NB	378,966	393,957	384,987	510,921	370,832	337,079	347,180	351,911	366,508	326,197	376,854
Apostle Islands NL	170,202	156,945	176,040	163,419	148,556	290,059	232,662	183,796	203,421	252,920	197,802
Appomattox Court House NHP	185,443	216,220	258,917	320,668	317,660	278,776	412,640	303,139	113,960	103,044	251,047
Arches NP	996,312	1,014,405	1,040,758	1,070,577	1,082,866	1,284,767	1,399,247	1,585,718	1,539,028	1,663,557	1,267,724
Arkansas Post NMEM	32,160	34,712	37,127	39,703	36,420	30,860	38,702	34,405	36,079	30,126	35,029
Arlington House The R.E. Lee MEM	603,773	627,576	576,816	562,772	586,531	671,638	532,050	1,057,500	726,059	49,500	599,422
Assateague Island NS	2,129,658	2,106,090	2,105,419	2,154,859	2,056,828	2,170,681	2,275,606	2,267,821	2,347,167	2,136,889	2,175,102
Aztec Ruins NM	38,234	37,437	41,106	44,744	41,312	44,721	53,165	57,692	52,756	54,933	46,610
Badlands NP	933,918	977,778	870,741	883,406	892,372	868,094	989,354	996,263	1,054,325	1,008,942	947,519
Bandelier NM	212,544	234,896	193,914	150,289	126,682	130,106	174,073	198,478	209,141	198,441	182,856
Belmont-Paul Women's Equality NM								2,704	10,893	9,081	7,559
Bent's Old Fort NHS	28,131	29,120	26,842	25,815	23,324	24,555	26,219	31,948	26,397	23,562	26,591
Bering Land Bridge NPRES	1,054	2,642	1,890	2,642	2,642	2,636	2,642	2,642	2,642	2,642	2,407
Big Bend NP	363,905	372,330	361,862	292,055	316,953	314,102	381,747	388,290	440,276	440,091	367,161
Big Cypress NPRES	812,207	665,523	941,393	882,570	1,007,788	1,192,856	1,112,290	1,102,148	922,883	794,350	943,401
Big Hole NB	49,822	44,771	36,290	35,207	38,092	36,987	39,549	41,795	36,717	56,943	41,617
Big South Fork NRRA	686,747	656,374	606,579	600,161	565,063	599,906	643,135	684,715	761,200	760,906	656,479
Big Thicket NPRES	100,509	140,489	137,722	135,262	120,108	109,956	125,832	192,809	170,649	221,514	145,485
Bighorn Canyon NRA	205,293	258,637	201,010	245,831	241,527	224,083	245,173	259,485	231,836	235,570	234,845
Biscayne NP	437,745	467,612	476,077	495,613	486,848	525,745	508,164	514,709	446,961	469,253	482,873
Black Canyon of the Gunnison NP	171,451	176,344	168,336	192,570	175,852	183,045	209,166	238,018	307,143	308,962	213,089
Blue Ridge PKWY	15,936,316	14,517,118	15,382,447	15,205,059	12,877,368	13,941,749	15,054,603	15,175,578	16,093,765	14,690,418	14,887,442
Bluestone NSR	45,904	37,790	41,670	36,842	36,266	38,450	36,733	34,139	32,028	31,316	37,114
Booker T. Washington NM	21,216	21,665	24,030	24,361	22,103	26,127	27,205	23,440	25,479	22,732	23,836

Annual Visitation By Park (1979

Image 10 – Figure contains the view of excel spreadsheet which was obtained form National Park Service. This is the unformatted original view of the obtained data.

	Park_name	Average	Year	Visitors
1	Abraham Lincoln Birthplace	206815	2009	221111
2	Acadia	2751707	2009	2227698
3	Adams	210647	2009	253656
4	African Burial Ground	71389	2009	*NA*
5	Agate Fossil Beds	14868	2009	12694
6	Alibates Flint Quarries	5160	2009	2918
7	Allegheny Portage Railroad	151311	2009	118931
8	Amistad	1385578	2009	2573966
9	Andersonville	125667	2009	136267
10	Andrew Johnson	50895	2009	63296
11	Aniakchak	87	2009	14
12	Antietam	376853	2009	378966
13	Apostle Islands	197802	2009	170202
14	Appomattox Court House	251046	2009	185443
15	Arches	1267723	2009	996312
16	Arkansas Post	35029	2009	32160
17	Arlington House The R.E. Lee	599421	2009	603773
18	Assateague Island	2175101	2009	2129658
19	Aztec Ruins	46610	2009	38234
20	Badlands	947519	2009	933918
21	Bandelier	182856	2009	212544
22	Belmont-Paul Women's Equality	7559	2009	*NA*

Showing 1 to 23 of 3,790 entries, 4 total columns

Image 11 – Figure contains the view of “annual_report” dataframe from R once the data has been appropriately cleaned up.

```r
dbGetQuery(db, "SELECT * FROM clean_parks")
```

R Console

data.frame
61 x 4

Name <chr>	State <chr>	Date_established <chr>	Area_acres <int>
Acadia	Maine	1919-02-26	4907526
American Samoa	American Samoa	1988-10-31	825667
Arches	Utah	1971-11-12	7667898
Badlands	South Dakota	1978-11-10	24275594
Big Bend	Texas	1944-06-12	80116321
Biscayne	Florida	1980-06-28	17297111
Black Canyon of the Gunnison	Colorado	1999-10-21	3077983
Bryce Canyon	Utah	1928-02-25	3583508
Canyonlands	Utah	1964-09-12	33759783
Capitol Reef	Utah	1971-12-18	24190450

1-10 of 61 rows Previous 1 2 3 4 5 6 7 Next

Image 12 – Figure contains the view of a SQL query from R. The query works as a demonstration that the tables were uploaded to the database and can be successfully retrieved.

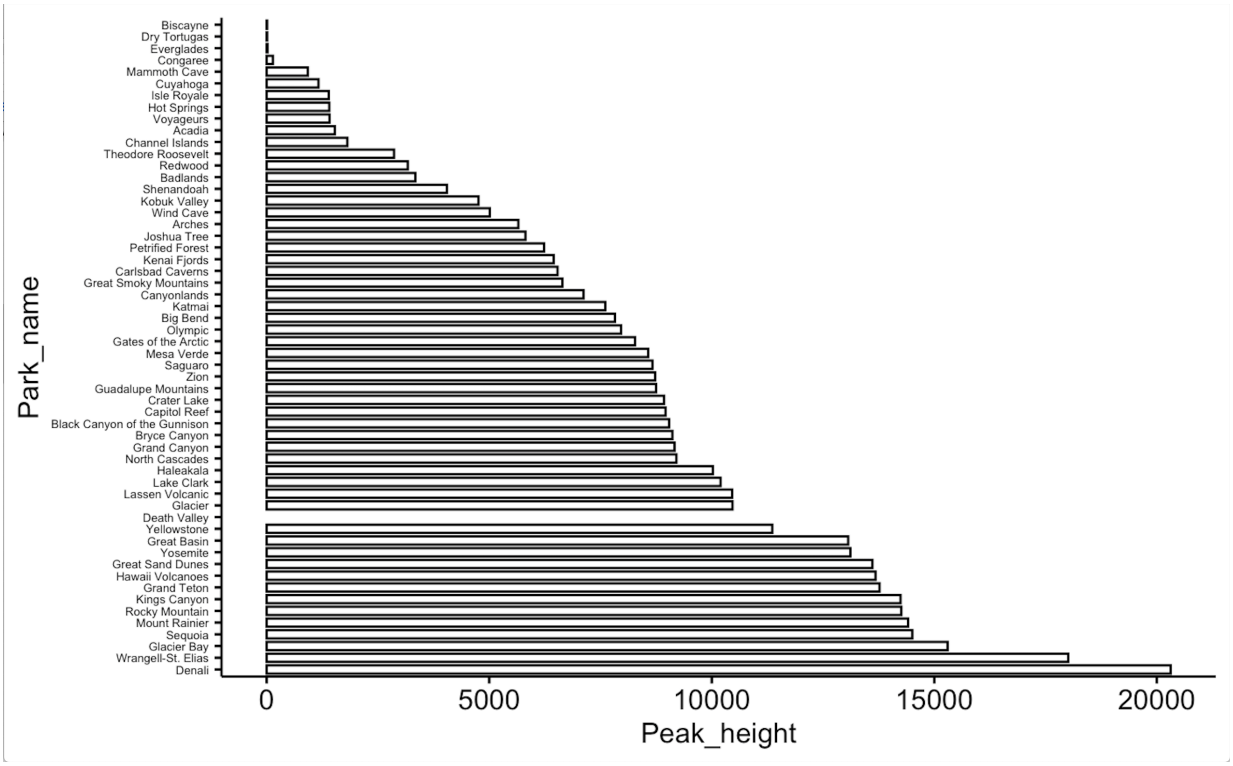


Image 13 – Figure illustrates the issue encountered with the formatted data from the open source. This figure demonstrates the presence of NA values for Death Valley in the "elev_all_parks" dataframe. The issue was successfully resolved.

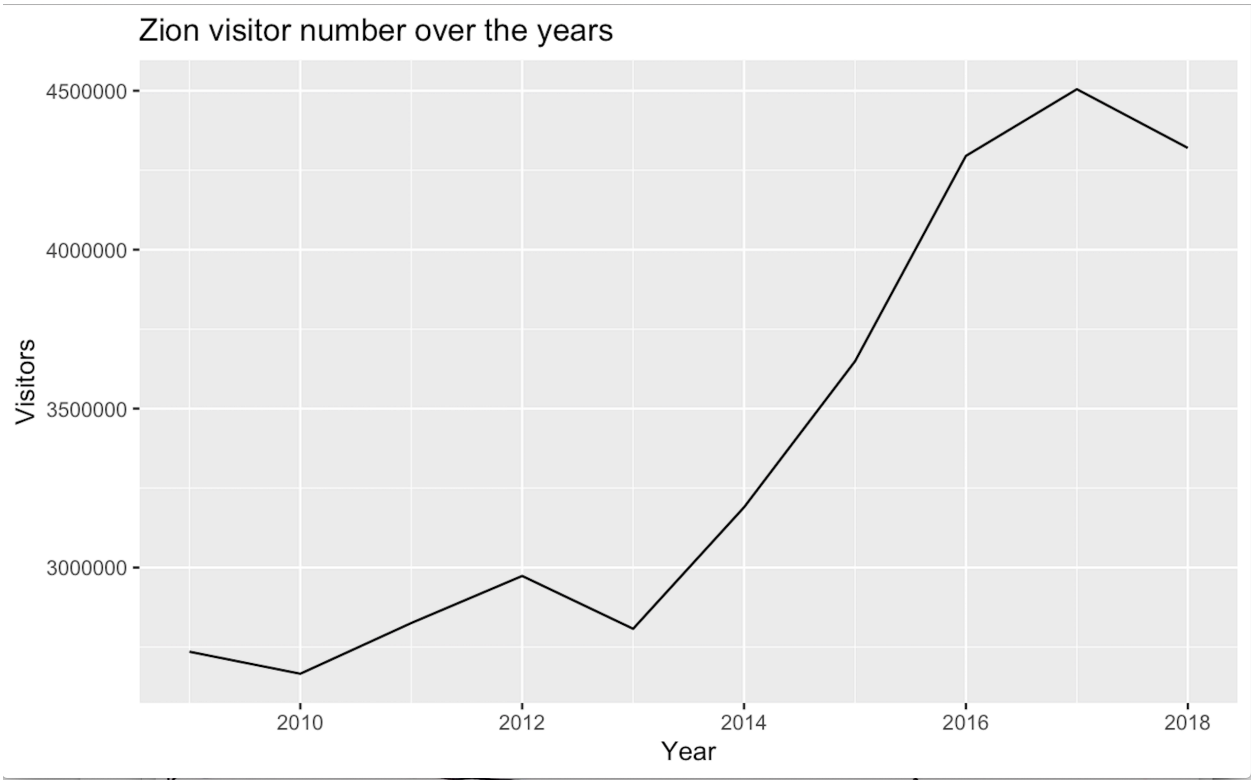


Image 14 – Figure demonstrates the change in the visitor number to Zion National Park over the years. This is an example of visualization and analysis that can be done with the extracted data.

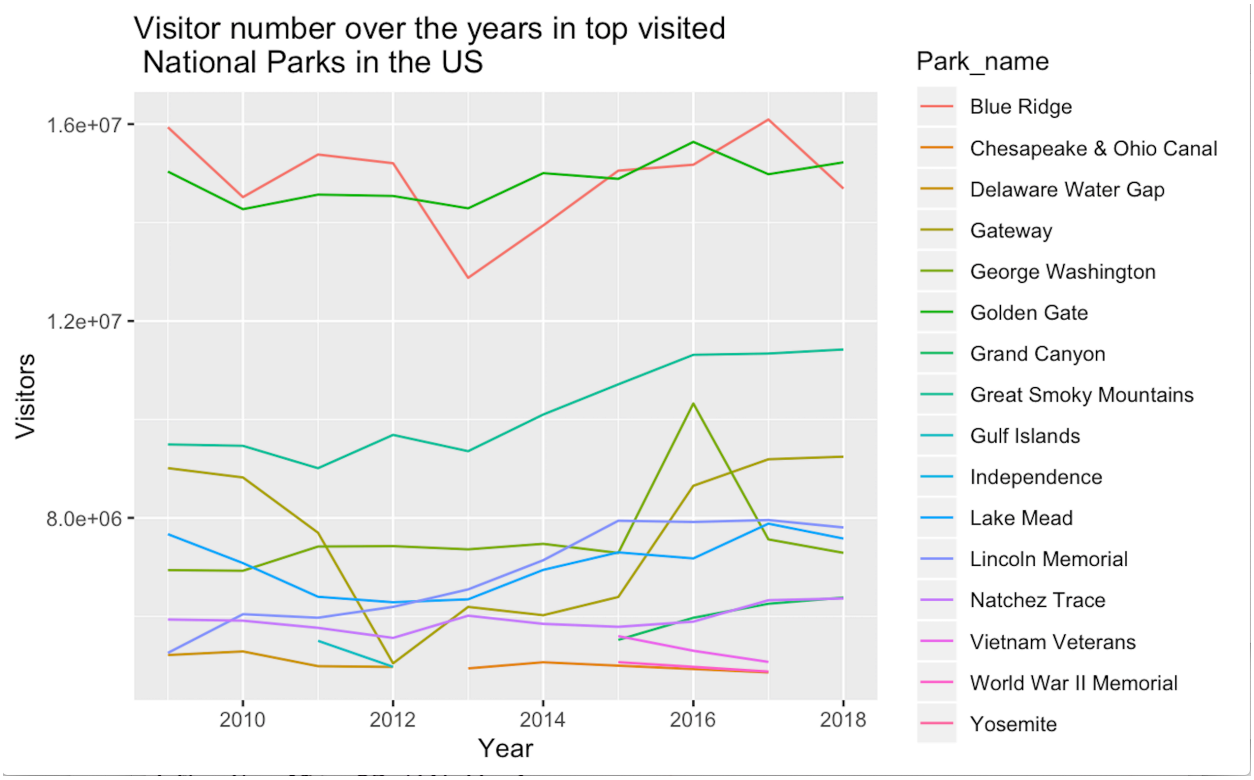


Image 15 - Demonstration of the most visited parks over the years, limited sample of data for this example plot.

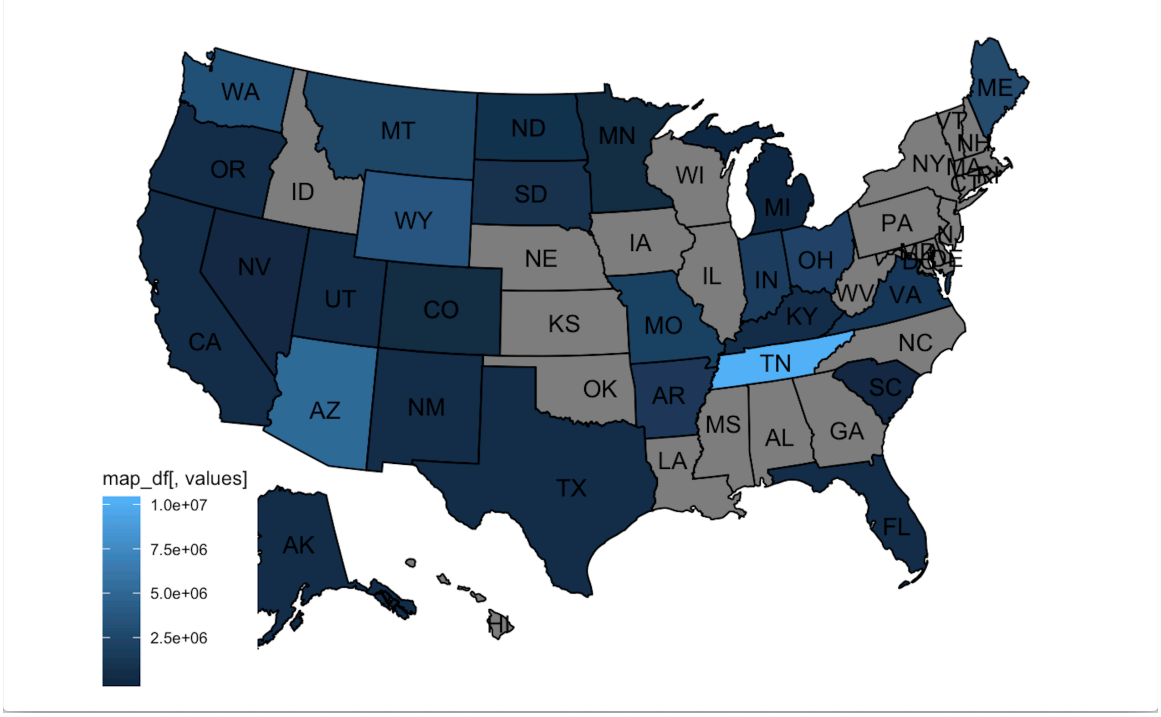


Image 16 – Figure depicts the density of available data of average visits to the national parks in the highlighted states (from dark blue to light blue). The average is over 10 year period for the recreational visits for the available national parks.

WORKS CITED

100 Years of Warming in the National Parks. 25 May 2016. Climate Central. <https://www.climatecentral.org/gallery/graphics/100-years-of-warming-at-the-national-parks>.

12 Day Hikes in Yellowstone. 4 June 2019. Yellowstone National Park Lodges. <https://www.yellowstonenationalparklodges.com/connect/yellowstone-hot-spot/trail-mix-12-awesome-day-hikes-in-yellowstone/>.

List of National Parks of the United States. n.d. Wikipedia. November 2019. <https://en.wikipedia.org/wiki/List_of_national_parks_of_the_United_States>.

National Parks Service. n.d. U.S. Department of the Interior. November 2019. <https://irma.nps.gov/STATS/Reports/Park>.

News Releases (U.S. National Park Service). n.d. National Parks Service. December 2019. <https://www.nps.gov/aboutus/news/news-releases.htm#sort=Date_Released desc&fq[]=Date_Released:[NOW-29DAYS TO NOW]>.

Olympic National Park Hiking Trails. n.d. National Parked. <https://www.nationalparked.com/olympic/hiking-trails.>.

Search and Rescue: Lessons from the Field. n.d. National Parks Service. <https://www.nps.gov/yose/blogs/psarblog.htm>.

Stats Report Viewer. n.d. National Parks Service. November 2019. <https://irma.nps.gov/STATS/SSRSReports/National Reports/Annual Visitation By Park (1979 - Last Calendar Year)>.

Visitation Numbers (U.S. National Park Service). n.d. National Parks Service. <https://www.nps.gov/aboutus/visitation-numbers.htm>.

Web Data Integration – Import.io - Data Extraction, Web Data Harvesting, Data Preparation, Data Integration. n.d. Import.io. <https://www.import.io/>.

Wee, Rolando Y. *Oldest National Parks In The United States.* 27 June 2016. WorldAtlas. November 2019. <https://www.worldatlas.com/articles/oldest-national-parks-in-the-united-states.html.>.

Western U.S. National Parks Popular with Foreign Tourists. n.d. Visa. <https://usa.visa.com/partner-with-us/visa-consulting-analytics/western-us-national-parks-popular-with-foreign-tourists.html>.

Yosemite Valley Day Hikes. n.d. National Parks Service. <https://www.nps.gov/yose/planyourvisit/valleyhikes.htm>.

Stoma_Project

Valentina Stoma

11/26/2019

```
library(rvest)
```

```
## Loading required package: xml2
```

```
library(stringr)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------- tidy
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v forcats 0.4.0
## v readr   1.3.1
```

```
## -- Conflicts ------------------------------------------------------------------------- tidyverse_
## x dplyr::filter()         masks stats::filter()
## x readr::guess_encoding() masks rvest::guess_encoding()
## x dplyr::lag()            masks stats::lag()
## x purrr::pluck()          masks rvest::pluck()
```

```
library(tidyr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(RSQLite)
library(sqldf)
```

```
## Loading required package: gsubfn
```

```
## Loading required package: proto
```

```
## Warning in doTryCatch(return(expr), name, parentenv, handler): unable to load shared object '/Library
##   dlopen(/Library/Frameworks/R.framework/Resources/modules//R_X11.so, 6): Library not loaded: /opt/X1
##   Referenced from: /Library/Frameworks/R.framework/Resources/modules//R_X11.so
##   Reason: image not found
```

```
## Could not load tcltk.  Will use slower R code instead.
```

```
library(readxl)
library(ggpubr)
```

```
## Loading required package: magrittr
```

```
##
## Attaching package: 'magrittr'
```

```
## The following object is masked from 'package:purrr':
##
```

```
##     set_names

## The following object is masked from 'package:tidyr':
##
##     extract
```

```
library(usmap)
library(ggplot2)
```

```
#reading in the files
parks<-read_csv("Wiki_parks.csv")
```

```
## Parsed with column specification:
## cols(
##   url = col_character(),
##   Name = col_character(),
##   Name_link = col_character(),
##   State = col_character(),
##   Date_established = col_character(),
##   Area = col_character()
## )
```

```
#deleting unneccessary columns that are an accessaory of using Import.io
parks<- parks[,-c(1,3)]
head(parks)
```

```
## # A tibble: 6 x 4
##   Name           State          Date_established  Area
##   <chr>          <chr>          <chr>             <chr>
## 1 Acadia         Maine          February 26, 1919 49,075.26 acres (198.6 km
## 2 American Samoa American Samoa October 31, 1988  8,256.67 acres (33.4 km
## 3 Arches         Utah           November 12, 1971 76,678.98 acres (310.3 km
## 4 Badlands       South Dakota   November 10, 1978 242,755.94 acres (982.4 km
## 5 Big Bend       Texas          June 12, 1944     801,163.21 acres (3,242.2 km
## 6 Biscayne       Florida        June 28, 1980     172,971.11 acres (700.0 km
```

```
#creating a separate column to contain the copy of the  Area column for further modifications
# and changing the format of the Date_estblished column into Date
parks <- parks %>%
  mutate(Area_km = Area) %>%
  mutate(Date_established = mdy(Date_established))

head(parks)
```

```
## # A tibble: 6 x 5
##   Name        State      Date_established Area               Area_km
##   <chr>       <chr>      <date>           <chr>              <chr>
## 1 Acadia      Maine      1919-02-26       49,075.26 acres (~ 49,075.26 acres (1~
## 2 American S~ American ~ 1988-10-31       8,256.67 acres (3~ 8,256.67 acres (33~
## 3 Arches      Utah       1971-11-12       76,678.98 acres (~ 76,678.98 acres (3~
## 4 Badlands    South Dak~ 1978-11-10       242,755.94 acres ~ 242,755.94 acres (~
## 5 Big Bend    Texas      1944-06-12       801,163.21 acres ~ 801,163.21 acres (~
## 6 Biscayne    Florida    1980-06-28       172,971.11 acres ~ 172,971.11 acres (~
```

```
#cleaning up the area column in order to obtain only the acres in one column
clean_parks<- parks %>%
  mutate(Area = str_replace(Area, "acres \\(", " ")) %>%
  mutate(Area = str_replace(Area, "km", "")) %>%
```

```
  separate(., Area, into=c("Area", "Area_new"), sep=" ")
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 61 rows [1, 2, 3, 4,
## 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```
#getting rid of unnecessary columns
clean_parks<-clean_parks[,-c(5,6)]
#My dplyr package is conflicting with some other packcages so for some of the
#functions I need to specify to which package it belongs.
clean_parks<- clean_parks %>%
  dplyr::rename(Area_acres = Area)
```

Issue arose with date format for the introduction into the SQL - needed to change it into charachter in the following manner in order to insert into SQL

```
#this format change allows me to have normal character format that will
#be read correctly in the SQL database
clean_parks$Date_established<- as.character(as.Date(as.character(clean_parks$Date_established),
                                                    format = "%Y-%m-%d"))

#Using this str_trim, I trim out any lagging and leading spaces, which prohibit recognition
#of the same names in different table in the database
#additionally, deleting any spaces in the acres size for proper formatting.
clean_parks <- clean_parks %>%
  mutate_if(is.character, str_trim) %>%
  mutate(Area_acres = str_replace(Area_acres, ",", ""))
#moreover, I am rounding all the acres sizing of the national parks for universal fomatting.
clean_parks<-clean_parks %>%
  mutate(Area_acres = str_replace_all(Area_acres, "[.*+]", ""))
```

10 MOST VISITED PARKS IN THE US

```
#reading in the website
visitors<- read_html("https://www.nps.gov/aboutus/visitation-numbers.htm")

#extracting names, visitors from the read in website
names <-visitors %>%
  html_nodes(".CS_Element_Custom~ .CS_Element_Custom+ .CS_Element_Custom thead+ tbody td:nth-child(2)") %>%
  html_text

number <- visitors %>%
  html_nodes("#cs_control_5546313 td~ td+ td") %>%
  html_text()


#combining the extracted information into a dataframe.
most_visits<- data.frame(names, number)
#clean up the creatde dataframe to have appropriate formats for number and characters,
#as well as formatting the numbers to not have any commas. For the universal notation in this project,
#the names of the National park do not contain any other identifications except its actual name.
most_visits<-most_visits %>%
  mutate(number = as.character(number)) %>%
  mutate(number = str_replace_all(number, ",", "")) %>%
  mutate(number = as.numeric(number)) %>%
```

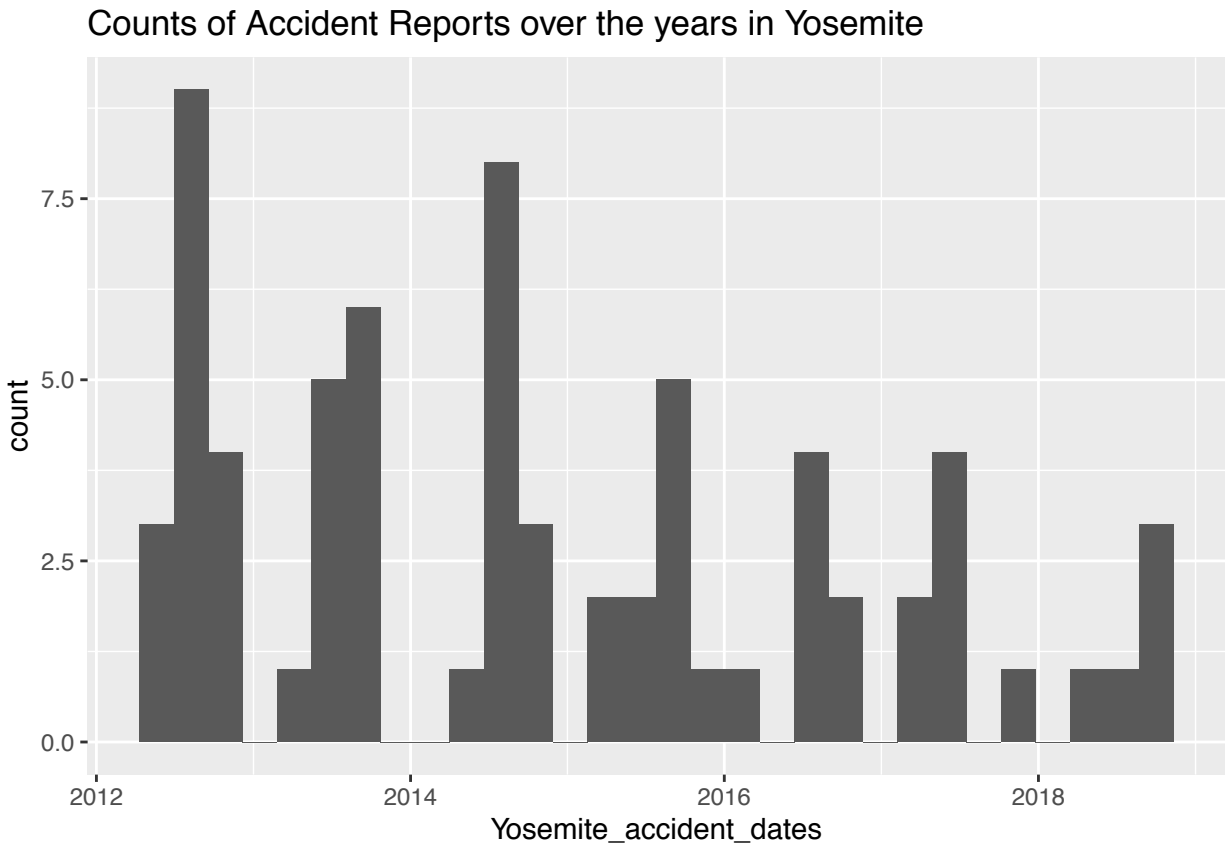
```
  mutate(names = str_replace(names, "National Park", "")) %>%
  dplyr::rename(visitor_number = number) %>%
  dplyr::rename(Park_name = names)
#Deleting any lagging and leading spaces.
most_visits <- most_visits %>%
  mutate_if(is.character, str_trim)
head(most_visits)
```

```
##               Park_name visitor_number
## 1 Great Smoky Mountains       15223697
## 2          Grand Canyon       14690418
## 3        Rocky Mountain       11421200
## 4                  Zion        9243305
## 5           Yellowstone        7804683
## 6              Yosemite        7578958
```

LIST OF YOSEMITE ACCIDENTS OVER THE YEARS

```
#reading in the web page with the accidents.
yosemite_accident<-read_html("https://www.nps.gov/yose/blogs/psarblog.htm")

#extracting the necessary information from the appropriate nodes.
Yosemite_accident_dates<-yosemite_accident %>%
  html_nodes(".date") %>%
  html_text()

description<-yosemite_accident %>%
  html_nodes(".slug p") %>%
  html_text()
#creating a list of just the names of the park these accidents refer to.
yose_name<-replicate(69, "Yosemite")
#this dataframe will not be included in the database, this is an example
#of missing data from other National Parks, and should be supplemented later on.
yose_acc<- data.frame(yose_name, Yosemite_accident_dates, description)
#modyifying the format of the date column as Date.
yose_acc<-yose_acc %>%
  mutate(Yosemite_accident_dates = mdy(Yosemite_accident_dates)) %>%
  dplyr::rename(accident_description = description)

#saving the dataframe to an appropriate name and modying the date format to be read into the
#SQL database in case in the future it will be supplemented with more national park accident data.
yosemite_accidents<- yose_acc
yosemite_accidents$Yosemite_accident_dates<- as.character(as.Date(as.character(yosemite_accidents$Yosem
                                                                               format = "%Y-%m-%d"))

#this plot is generated to demostrate when the accidents occur and to show that
#the data can be used in the analysis if more is collected accessibly by the national parks.
ggplot(yose_acc, aes(Yosemite_accident_dates)) +
  geom_histogram() +
  ggtitle("Counts of Accident Reports over the years in Yosemite")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



YELLOWSTONE TRAIL NAME and DIFFICULTY INFORMATION

```
#reading the web site containing the trail information.
yellowstone_trail<-read_html("https://www.yellowstonenationalparklodges.com/connect/yellowstone-hot-spo

#findign and reading in the nodes that have the inforamtion about names and trail difficulty.
trail_names<-yellowstone_trail %>%
  html_nodes("h3 strong") %>%
  html_text()

yellowstone_diff<- yellowstone_trail %>%
  html_nodes("p:nth-child(18) , p:nth-child(23) , p:nth-child(34) , p:nth-child(39) , p:nth-child(44) ,
  html_text()


yellow_loc<- yellowstone_trail %>%
  html_nodes("p:nth-child(65) , p:nth-child(60) , p:nth-child(55) , p:nth-child(50) , p:nth-child(45) ,
  html_text()
#creating a list containig the name of the national park for which I obtained the trail
#information. 12 is the number of trails I found from one source.
yellow_name<-replicate(12, "Yellowstone")
yellow_trails<- data.frame(yellow_name,trail_names, yellowstone_diff, yellow_loc)

#changing the content of one of the columns because of the way that the nodes read into R
```

```
yellow_trails <- yellow_trails %>%
  mutate(yellowstone_diff = str_replace(yellowstone_diff, "Location: Canyon Area", "Level of Difficulty
#cleaning up the obtained dataframe to reduce text in the level of difficulty column
#and to standardize the classification of trail level. Additioanlly, changing the names
#of the columns to accomodate addition of other national park trail information.
yellowstone_trails <- yellow_trails %>%
  mutate(yellowstone_diff = str_replace(yellowstone_diff, "Level of [Dd]ifficulty:", "")) %>%
  mutate(yellowstone_diff = str_replace(yellowstone_diff, "Difficulty:", "")) %>%
  dplyr::rename(difficulty_level = yellowstone_diff) %>%
  mutate(trail_names = str_to_sentence(trail_names)) %>%
  mutate(yellow_loc = str_replace(yellow_loc, "Location: ", "")) %>%
  dplyr::rename(trail_location = yellow_loc) %>%
  dplyr::rename(park_name = yellow_name) %>%
  dplyr::rename(yellowstone_trail_name = trail_names)
```

Elevation of parks

```
#this dataframe was obtained from Import.io
#there is some missing values in the Mountain range column in the dataframe so just in case replcae it
eleva_w_parks<- read.csv("Elevation_wiki_parks-(Crawl-Run)---2019-11-30T145634Z.csv", na.strings=c("","
head(eleva_w_parks)
```

```
##            Park_name         Peak_name        Mountain_range
## 1             Denali            Denali          Alaska Range
## 2 Wrangell-St. Elias Mount Saint Elias Saint Elias Mountains
## 3        Glacier Bay Mount Fairweather Saint Elias Mountains
## 4            Sequoia     Mount Whitney         Sierra Nevada
## 5      Mount Rainier     Mount Rainier         Cascade Range
## 6     Rocky Mountain        Longs Peak           Front Range
##          Mim_elevation       Vertical.relief
## 1      240 feet (73 m) 20,070 feet (6,120 m)
## 2        0 feet (0 m) 18,008 feet (5,489 m)
## 3        0 feet (0 m) 15,300 feet (4,700 m)
## 4   1,360 feet (410 m) 13,145 feet (4,007 m)
## 5   1,610 feet (490 m) 12,801 feet (3,902 m)
## 6 7,630 feet (2,330 m)  6,629 feet (2,021 m)
```

```
#Formatting the type of columns in the dataframe.
elevation<-eleva_w_parks %>%
  mutate(Mim_elevation = as.character(Mim_elevation)) %>%
  mutate(Vertical.relief = as.character(Vertical.relief))
elevation= as.tibble(elevation)
```

```
## Warning: `as.tibble()` is deprecated, use `as_tibble()` (but mind the new semantics).
## This warning is displayed once per session.
```

```
#renaming the column because fof the typo from Import.io
elevation<-elevation %>%
  dplyr::rename(Min_elevation = Mim_elevation)

head(elevation)
```

```
## # A tibble: 6 x 5
##   Park_name     Peak_name     Mountain_range   Min_elevation   Vertical.relief
```

```
##   <fct>          <fct>          <fct>            <chr>           <chr>
## 1 Denali         Denali         Alaska Range     240 feet (73 m) 20,070 feet (6,1~
## 2 Wrangell-St.~ Mount Saint ~ Saint Elias Mou~ 0 feet (0 m)    18,008 feet (5,4~
## 3 Glacier Bay    Mount Fairwe~ Saint Elias Mou~ 0 feet (0 m)    15,300 feet (4,7~
## 4 Sequoia        Mount Whitney Sierra Nevada    1,360 feet (41~ 13,145 feet (4,0~
## 5 Mount Rainier Mount Rainier Cascade Range    1,610 feet (49~ 12,801 feet (3,9~
## 6 Rocky Mounta~ Longs Peak     Front Range      7,630 feet (2,~ 6,629 feet (2,02~
```

We can calculate the final elevation at the peak level from the provided min_elevation (ground level for the park) and the vertical.relief (vertical chanhe experienced by the park).

```
#cleaning up the data further to only contain one column with clean values of
#min_elevation and vertical.relief  in feet
elev<- elevation %>%
  mutate(Min_elevation = str_replace_all(Min_elevation, "feet", " " )) %>%
  mutate(Min_elevation = str_replace(Min_elevation, "m", "")) %>%
  mutate(Min_elevation = str_replace(Min_elevation, "\\(", "")) %>%
  mutate(Min_elevation = str_replace(Min_elevation, "\\)", "")) %>%
  separate(., Min_elevation, into = c("Min_elevation_feet", "Min_elevation_m"),
           sep = " ", remove = TRUE)
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 56 rows [1, 2, 3, 4,
## 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```
elev<- elev[, -c(5)]

elev<-elev %>%
  mutate(Vertical.relief = str_replace_all(Vertical.relief, "feet", " " )) %>%
  mutate(Vertical.relief = str_replace(Vertical.relief, "m", "")) %>%
  mutate(Vertical.relief = str_replace(Vertical.relief, "\\(", "")) %>%
  mutate(Vertical.relief = str_replace(Vertical.relief, "\\)", "")) %>%
  separate(., Vertical.relief, into= c("Vertical_relief_feet", "Vertical.relief.m"),
           sep = " ", remove = TRUE)
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 56 rows [1, 2, 3, 4,
## 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```
elevation_all_parks<- elev[, -c(6)]
head(elevation_all_parks)
```

```
## # A tibble: 6 x 5
##   Park_name    Peak_name    Mountain_range    Min_elevation_f~ Vertical_relief_~
##   <fct>        <fct>        <fct>             <chr>            <chr>
## 1 Denali       Denali       Alaska Range      240              20,070
## 2 Wrangell-St~ Mount Saint~ Saint Elias Moun~ 0                18,008
## 3 Glacier Bay  Mount Fairw~ Saint Elias Moun~ 0                15,300
## 4 Sequoia      Mount Whitn~ Sierra Nevada     1,360            13,145
## 5 Mount Raini~ Mount Raini~ Cascade Range     1,610            12,801
## 6 Rocky Mount~ Longs Peak   Front Range       7,630            6,629
```

Calculating the final elevation :

```
#finally, calcualting the final elevation level for the peaks from the presented data.
elev_all_parks<-elevation_all_parks %>%
  mutate(Min_elevation_feet = str_replace_all(Min_elevation_feet, ",", "")) %>%
  mutate(Vertical_relief_feet = str_replace_all(Vertical_relief_feet, ",", "")) %>%

  mutate(Min_elevation_feet = as.integer(Min_elevation_feet)) %>%
```

```
  mutate(Vertical_relief_feet = as.integer(Vertical_relief_feet)) %>%
  mutate(Peak_height = Min_elevation_feet + Vertical_relief_feet)
```

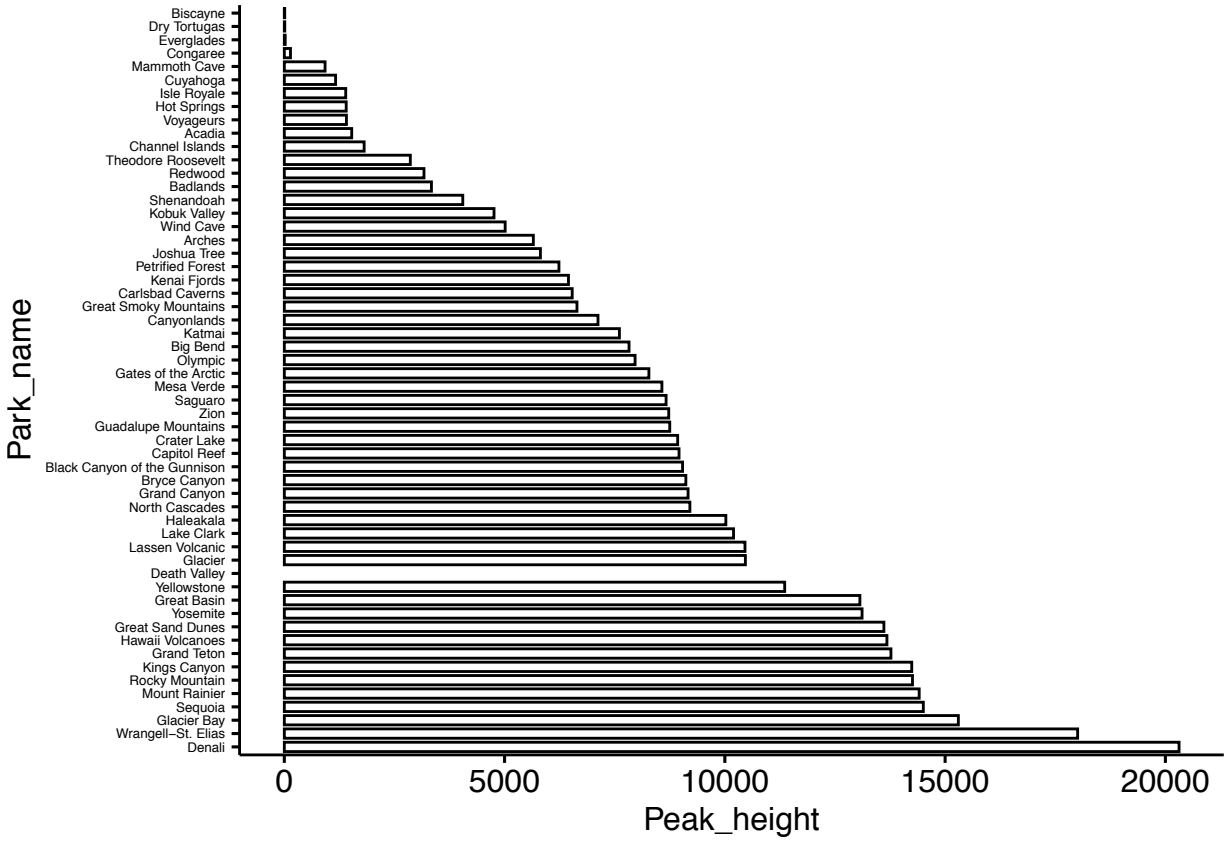
```
## Warning: NAs introduced by coercion
```

```
#final clean up of the datafram to contain the correct formatting, as well as
#trim away any leading and lagging spaces.
elev_all_parks<- elev_all_parks %>%
  mutate(Park_name = as.character(Park_name)) %>%
  mutate(Peak_name = as.character(Peak_name)) %>%
  mutate(Mountain_range = as.character(Mountain_range)) %>%
  mutate_if(is.character, str_trim)
```

Plot, which demostrates the peak height for the provided national parks

```
#plotting the range of peaks for all the national parks. Changing the size of the
#text for the ticks in order for them to not overlap.
peaks_plot<-ggbarplot(elev_all_parks, x = "Park_name", y = "Peak_height") +
  coord_flip()
peaks_plot + theme(axis.text.y =element_text(size = 5))
```

```
## Warning: Removed 1 rows containing missing values (position_stack).
```



YOSEMITE TRAIL INFORMATION

```
#additional trail information from a different national park - Yosemite.
#The website is read with rvest.
```

```
yose_trails<-read_html("https://www.nps.gov/yose/planyourvisit/valleyhikes.htm")

yose_trail_names<- yose_trails %>%
  html_nodes("#cs_idCell2x1x1 a") %>%
  html_text()

diff_yose_trail<- yose_trails %>%
  html_nodes("tr+ tr td:nth-child(2)") %>%
  html_text()

#similar to the previous
yose_name<-replicate(10, "Yosemite")
yosemite_trails<- data.frame(yose_name, yose_trail_names, diff_yose_trail)
```

Connecting the three dataframes to contain the trail and difficulty information.

```
#cleaning up the final dataframes in order to connect them into one trail information containing datafr
yellow_trail_connect<-yellowstone_trails[-4]
yellow_trail_connect <- as.tibble(yellow_trail_connect)

#making sure that the names are the same as in all other tibbles to connect into one dataframe
yellow_trail_connect <- yellow_trail_connect %>%
  dplyr::rename(park = park_name) %>%
  dplyr::rename(trail_name = yellowstone_trail_name) %>%
  dplyr::rename(difficulty = difficulty_level)



yosemite_trails_connect<- yosemite_trails %>%
  dplyr::rename(park = yose_name) %>%
  dplyr::rename(trail_name = yose_trail_names) %>%
  dplyr::rename(difficulty = diff_yose_trail )
```

```
#connecting the dataframes
trail_level<-rbind(yellow_trail_connect, yosemite_trails_connect)
head(trail_level)
```

```
## # A tibble: 6 x 3
##   park        trail_name                              difficulty
##   <fct>       <chr>                                   <chr>
## 1 Yellowstone Storm point loop trail                  " Easy"
## 2 Yellowstone Trout lake                              " Easy"
## 3 Yellowstone Mystic falls                            " Easy"
## 4 Yellowstone Tower fall overlook                     " Easy"
## 5 Yellowstone Uncle tom's trail (from artist point) " Moderate"
## 6 Yellowstone Elephant back mountain trail            " Moderate"
```

```
#the third data subset to add to the final trail iformation containing dataframe
olympic_trails<-read_csv("nationalparked-olympic-data-(Crawl-Run)---2019-12-01T191803Z.csv")
```

```
## Parsed with column specification:
## cols(
##   url = col_character(),
##   trail_name = col_character(),
##   difficulty = col_character()
## )
```

```
olympic_trails<- olympic_trails[,-c(1)]
olymp_names<-replicate(9, "Olympic")
olympic_trails_connect<-data.frame(olymp_names, olympic_trails)
olympic_trails_connect <- olympic_trails_connect %>%
  dplyr::rename(park = olymp_names)
head(olympic_trails_connect)
```

```
##      park                    trail_name       difficulty
## 1 Olympic   Ancient Groves Nature Trail             Easy
## 2 Olympic                Hall of Mosses             Easy
## 3 Olympic Hurricane Ridge Meadow Trails Easy to Moderate
## 4 Olympic           Madison Creek Falls        Very Easy
## 5 Olympic                Marymere Falls  Moderately Easy
## 6 Olympic                  Rialto Beach         Moderate
```

```
#adding the third data subset to the previosuly made up dataframe with trail information
trail_level_parks<- rbind(trail_level, olympic_trails_connect)
#figuring out which difficulty levels need to be fomratted
trail_level_parks %>%
  group_by(difficulty) %>%
  count()
```

```
## # A tibble: 10 x 2
## # Groups:   difficulty [10]
##    difficulty                n
##    <chr>                 <int>
##  1 " Easy"                   5
##  2 " Moderate"               4
##  3 " Strenuous"              3
##  4 Easy                      6
##  5 Easy to Moderate          2
##  6 Moderate                  3
##  7 Moderate to Strenuous     2
##  8 Moderately Easy           2
##  9 Strenuous                 3
## 10 Very Easy                 1
```

```
#formatting the difficulty level to contain universal names
trail_level_3<-trail_level_parks %>%
  mutate(difficulty = str_replace_all(difficulty, " ", "")) %>%
  mutate(difficulty = str_replace_all(difficulty, "ModeratelyEasy", "EasytoModerate")) %>%
  mutate(difficulty = str_replace_all(difficulty, "VeryEasy", "Easy")) %>%
  mutate(difficulty = str_replace_all(difficulty, "EasytoModerate", "Moderate")) %>%
  mutate(difficulty = str_replace_all(difficulty, "ModeratetoStrenuous", "Strenuous"))
#doubel checking that there are only desired level of difficulty.
trail_level_3 %>%
  group_by(difficulty) %>%
  count()
```

```
## # A tibble: 3 x 2
## # Groups:   difficulty [3]
##   difficulty     n
##   <chr>      <int>
## 1 Easy          12
## 2 Moderate      11
```

```
## 3 Strenuous      8
```

```
trail_level_3<- trail_level_3 %>%
  mutate_if(is.character, str_trim)
head(trail_level_3)
```

```
## # A tibble: 6 x 3
##   park        trail_name                              difficulty
##   <fct>       <chr>                                   <chr>
## 1 Yellowstone Storm point loop trail                  Easy
## 2 Yellowstone Trout lake                              Easy
## 3 Yellowstone Mystic falls                            Easy
## 4 Yellowstone Tower fall overlook                     Easy
## 5 Yellowstone Uncle tom's trail (from artist point) Moderate
## 6 Yellowstone Elephant back mountain trail          Moderate
```

Oldest parks:

can compare with the previous publicly available data.

```
#Reading in the web page with the 10 oldest parks in order to compare
#it to the wikipedia provided information
oldest<-read_html("https://www.worldatlas.com/articles/oldest-national-parks-in-the-united-states.html")

#extracting the nodes with the necessary information.
oldest_names<- oldest %>%
  html_nodes("td:nth-child(2)") %>%
  html_text()

date_oldest<- oldest %>%
  html_nodes("td~ td+ td") %>%
  html_text()
#uniting as a dataframe.
old_parks<- data.frame(oldest_names, date_oldest)
#cleaning up the dataframe to separate the park and the state information into different columns.
old_parks<-old_parks %>%
  separate(., oldest_names, into = c("Park","State"), sep = ",")
#cleaning up the dates column in order to be able to format it into the date format.
old_park_2<- old_parks %>%
  mutate(State = str_replace(State,"Wyoming-Montana-", "")) %>%
  mutate(date_oldest = str_replace_all(date_oldest, "st|th|nd", "")) %>%
  mutate(date_oldest = mdy(date_oldest))

#performing the same format change in order to upload it into the SQL database.
old_park_2$date_oldest<- as.character(as.Date(as.character(old_park_2$date_oldest), format = "%Y-%m-%d")
#trimming out the leading and lagging spaces.
old_park<-old_park_2 %>%
  mutate_if(is.character, str_trim)
head(old_park_2)
```

```
##            Park         State date_oldest
## 1   Yellowstone         Idaho  1872-03-01
## 2       Sequoia    California  1890-09-25
## 3      Yosemite    California  1890-10-01
## 4 Mount Rainier    Washington  1899-03-02
```

```
## 5   Crater Lake        Oregon  1902-05-22
## 6     Wind Cave  South Dakota  1903-01-09
```

RECREATIONAL VISITS FOR 10 YEARS

```
#reading in the obtained annual visitations report.
annual_report<-read_excel("Annual_park_1979_now.xlsx", col_names = TRUE, skip = 6 )
```

```
## New names:
## * `` -> ...1
## * `` -> ...2
## * `` -> ...3
## * `` -> ...4
## * `` -> ...5
## * ... and 9 more problems
```

```
#cleaning up the unnecesasry artificats of reading in the excel file into R
annual_report<-annual_report[-1, -c(11,12)]
#performing this name change for now to keep easy track of changed instead of numbers.
colnames(annual_report)<- c("Park_name", "one", "two", "three", "four", "five", "six", "sev", "eight",
```

```
#changing the layout and the structure of the dataframe to assign different year
#values to the rows instead of the seprate columns
annual_report<- gather(annual_report, Year, Visitors, one:tin)
#performign the name change back tot he years
annual_report <- annual_report %>%
  mutate(Year = str_replace(Year, "one", "2009")) %>%
  mutate(Year = str_replace(Year, "two", "2010"))  %>%
  mutate(Year = str_replace(Year, "three", "2011"))  %>%
  mutate(Year = str_replace(Year, "four", "2012"))  %>%
  mutate(Year = str_replace(Year, "five", "2013"))  %>%
  mutate(Year = str_replace(Year, "six", "2014"))  %>%
  mutate(Year = str_replace(Year, "sev", "2015"))  %>%
  mutate(Year = str_replace(Year, "eight", "2016"))  %>%
  mutate(Year = str_replace(Year, "nine", "2017"))  %>%
  mutate(Year = str_replace(Year, "tin", "2018"))

#formatting the values in the Average column to read as integer and round it as
#some of the values have too many significant values
annual_report<- annual_report %>%
  mutate(Average = as.integer(Average)) %>%
  mutate(Average = format(round(Average, 2), nsmall = 2))
```

```
#formatting the types of columns
annual_report$Year <- as.numeric(annual_report$Year)
annual_report$Average <- as.numeric(annual_report$Average)
head(annual_report)
```

```
## # A tibble: 6 x 4
##   Park_name                      Average  Year Visitors
##   <chr>                            <dbl> <dbl>    <dbl>
## 1 Abraham Lincoln Birthplace NHP  206815  2009   221111
## 2 Acadia NP                      2751707  2009  2227698
## 3 Adams NHP                       210647  2009   253656
## 4 African Burial Ground NM         71389  2009       NA
```

```
## 5 Agate Fossil Beds NM             14868  2009    12694
## 6 Alibates Flint Quarries NM        5160  2009     2918
```

```
#modifying the internal formatting of the names of the parks to make sure
#that the names are recognizable in different table in the SQL database.
annual_report <- annual_report %>%
  mutate(Park_name = str_replace_all(Park_name, "NHP|NM|NP|NHS|NRA|EM|NS|MEM|RES|NNRA|NSR|NS|NRES|NB|& P
  mutate(Park_name = str_replace_all(Park_name, "Shiloh P", "Shiloh")) %>%
  mutate(Park_name = str_replace_all(Park_name, "Pea Ridge P", "Pea Ridge")) %>%
  mutate(Park_name = str_replace_all(Park_name, "Ozark R", "Ozark")) %>%
  mutate(Park_name = str_replace_all(Park_name, "Niobrara R", "Niobrara")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Chickamauga & Chattanooga P", "Chickamauga & Chattanooga
  mutate(Park_name = str_replace_all(Park_name, "Fredericksburg & Spotsylvania P", "Fredericksburg & Spo
  mutate(Park_name = str_replace_all(Park_name, "Gettysburg P", "Gettysburg")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Guilford Courthouse P", "Guilford Courthouse"))%>%
  mutate(Park_name = str_replace_all(Park_name,"Horseshoe Bend P", "Horseshoe Bend")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Kennesaw Mountain P" ,"Kennesaw Mountain")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Kings Mountain P", "Kings Mountain")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Manassas P",  "Manassas"))%>%
  mutate(Park_name = str_replace_all(Park_name,"Richmond P", "Richmond" ))%>%
  mutate(Park_name = str_replace_all(Park_name,"River Raisin P", "River Raisin")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Saint Croix R" , "Saint Croix"))%>%
  mutate(Park_name = str_replace_all(Park_name,"S&RR" , "")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Vicksburg P", "Vicksburg"))
```

```
#trim out all the unnecessary lagging and leading spaces that prevent
#tables inteh database from joining on the same values of the park name.
annual_report<-annual_report %>%
  mutate_if(is.character, str_trim)
head(annual_report)
```

```
## # A tibble: 6 x 4
##   Park_name                   Average  Year Visitors
##   <chr>                         <dbl> <dbl>    <dbl>
## 1 Abraham Lincoln Birthplace  206815  2009   221111
## 2 Acadia                     2751707  2009  2227698
## 3 Adams                       210647  2009   253656
## 4 African Burial Ground        71389  2009       NA
## 5 Agate Fossil Beds            14868  2009    12694
## 6 Alibates Flint Quarries       5160  2009     2918
```

Example of the kind of visualisation that can be used to address the chnages in the visitor number for different parks

```
#plot example of the analysis that can be done on visito number to certain parks for all the years.
zio_plot<-annual_report %>%
  filter(Park_name == "Zion") %>%
  ggplot() + geom_line(aes(x = Year, y = Visitors )) +
  ggtitle("Zion visitor number over the years")
```

```
#subsetting the data arbitrary to only contain the to 100 most visited
#parks irrespective of the park and year limitations - done for exampel visualizaztion purposes.
plot100_most<-annual_report %>%
  group_by(Year) %>%
  arrange(desc(Visitors)) %>%
  head(100)
```

Demonstration of the most visited parks over the years, limited sample of data for this example plot.

```
#another plot as an example of the type of visualization that can be done on the changes in
#visitors to the parks - in this case subsetting the data arbitrary at 100 most visited parks
#all through out the ten year reported period, Becasue of this arbitrary cut off, some parks
#only have sporadic values, as they reached the significant level
top100_plot<-ggplot(plot100_most, aes(x = Year, y = Visitors, color = Park_name)) +
  geom_line() +
  ggtitle("Visitor number over the years in top visited\n National Parks in the US ")
```

Initate the database creation

```
db <- dbConnect(SQLite(), dbname="NationalParkInfo.sqlite")
dbSendQuery(conn = db, "pragma foreign_keys=on;")
```

```
## <SQLiteResult>
##   SQL  pragma foreign_keys=on;
##   ROWS Fetched: 0 [complete]
##        Changed: 0
```

Clean parks

```
#creating tables, identifying the primary key
# dbSendQuery(conn = db,"CREATE TABLE clean_parks (
#                         Name TEXT,
#                         State TEXT,
#                         Date_established DATE,
#                         Area_acres INTEGER,
#                         PRIMARY KEY (Name))
#                         WITHOUT ROWID")
```

```
#inserting the data from the appropriate tables in to the
# dbWriteTable(conn = db, name = "clean_parks", value = clean_parks, row.names=FALSE, append = TRUE)
```

```
#example of the equery working on the provided data -
clean_park_q<-dbGetQuery(db, "SELECT * FROM clean_parks")
```

```
## Warning: Closing open result set, pending rows
```

```
## Warning in result_fetch(res@ptr, n = n): Column `Area_acres`: mixed type, first
## seen values of type integer, coercing other values of type string
```

```
head(clean_park_q)
```

```
##             Name          State Date_established Area_acres
## 1         Acadia          Maine       1919-02-26    4907526
## 2 American Samoa American Samoa       1988-10-31     825667
## 3         Arches           Utah       1971-11-12    7667898
## 4       Badlands   South Dakota       1978-11-10   24275594
## 5       Big Bend          Texas       1944-06-12   80116321
## 6       Biscayne        Florida       1980-06-28   17297111
```

Annual Report

```
# dbSendQuery(conn = db,"CREATE TABLE annual_report (
#                         Park_name TEXT,
#                         Average Integer,
#                         Year INTEGER,
#                         Visitors INTEGER,
```

```
#                         PRIMARY KEY (Park_name, Year))
#                         WITHOUT ROWID")
```

```
# dbWriteTable(conn = db, name = "annual_report", value = annual_report, row.names=FALSE, append = TRUE
```

Visist

```
# dbSendQuery(conn = db,"CREATE TABLE most_visits  (
#                         Park_name TEXT,
#                         visitor_number INTEGER,
#                         PRIMARY KEY (Park_name))
#                         WITHOUT ROWID")
```

```
# dbWriteTable(conn = db, name = "most_visits", value = most_visits, row.names=FALSE, append = TRUE)
```

Elevation

```
  # dbSendQuery(conn = db,"CREATE TABLE elev_all_parks (
  #                         Park_name TEXT,
  #                         Peak_name TEXT,
  #                         Mountain_range TEXT,
  #                         Min_elevation_feet INTEGER,
  #                         Vertical_relief_feet INTEGER,
  #                         Peak_height INTEGER,
  #                         PRIMARY KEY (Park_name))
  #                         WITHOUT ROWID")
```

```
# dbWriteTable(conn = db, name = "elev_all_parks", value = elev_all_parks, row.names=FALSE, append = TR
```

Trail difficulty

```
# dbSendQuery(conn = db,"CREATE TABLE trail_level_3 (
#                       park TEXT,
#                       trail_name TEXT,
#                       difficulty TEXT,
#                       PRIMARY KEY (trail_name))
#                       WITHOUT ROWID")
```

```
# dbWriteTable(conn = db, name = "trail_level_3", value = trail_level_3, row.names=FALSE, append = TRUE
```

```
# dbSendQuery(conn = db,"CREATE TABLE old_park_2 (
#                       Park TEXT,
#                       State TEXT,
#                       date_oldest DATE,
#                       PRIMARY KEY (Park))
#                       WITHOUT ROWID")
```

```
#dbWriteTable(conn = db, name = "old_park_2", value = old_park_2, row.names=FALSE, append = TRUE)
```

```
#visualization of the tables in the database.
dbListTables(db)
```

```
## [1] "annual_report"  "clean_parks"    "elev_all_parks" "most_visits"
## [5] "old_park_2"     "trail_level_3"
```

```
#This gets the visitors number for 2009 from the National Park Services
#for only national parks in the US reported by the Wikipedia
all_visits_2009<-dbGetQuery(db, "SELECT annual_report.Park_name, annual_report.Visitors, annual_report.
          FROM annual_report
```

```
          INNER JOIN clean_parks on clean_parks.Name = annual_report.Park_name
          WHERE annual_report.Year = 2009")
head(all_visits_2009)
```

```
##                       Park_name Visitors Year
## 1                        Acadia  2227698 2009
## 2                        Arches   996312 2009
## 3                      Badlands   933918 2009
## 4                      Big Bend   363905 2009
## 5                      Biscayne   437745 2009
## 6 Black Canyon of the Gunnison   171451 2009
```

```
#this shows that the names are the same in the annual_report and clean_parks,
#and that SQL queries will work on these tables when they need to be joined.
same_name<-dbGetQuery(db, "SELECT annual_report.Park_name FROM annual_report INTERSECT SELECT clean_par
head(same_name)
```

```
##                       Park_name
## 1                        Acadia
## 2                        Arches
## 3                      Badlands
## 4                      Big Bend
## 5                      Biscayne
## 6 Black Canyon of the Gunnison
```

```
#This demonstrates that the names are identical in other dataframes as well.
old_park_query<-dbGetQuery(db, "SELECT old_park_2.park FROM old_park_2 INTERSECT SELECT annual_report.Pa
head(old_park_query)
```

```
##             Park
## 1    Crater Lake
## 2        Glacier
## 3      Haleakala
## 4     Mesa Verde
## 5  Mount Rainier
## 6 Rocky Mountain
```

```
library(maps)
```

```
##
## Attaching package: 'maps'
```

```
## The following object is masked from 'package:purrr':
##
##     map
```

```
usa <- map_data("usa")
```

```
#creating a separate column that has the abbreaviation of the states in order to be able to plot it wit
clean_parks_abb<- clean_parks
clean_parks_abb$State_abb<-state.abb[match(clean_parks_abb$State,state.name)]

plot_us_clean<-clean_parks_abb %>%
  dplyr::rename(State_name = State)  %>%
  dplyr::rename(state = State_abb)
```

Connect with annual report with clean data info and plot the visitor population.

```
#I used this quaery as an example for future work that this database can be used for -
#specifically, verification of the data obtained from open web soruces. T
#his cross reference verification is can be used with dates and other data.
compare_dates<-dbGetQuery(db, "SELECT clean_parks.Name, clean_parks.State, clean_parks.Date_established
           FROM clean_parks
            INNER JOIN old_park_2 on old_park_2.Park = clean_parks.Name ")
head(compare_dates)
```

```
##             Name      State Date_established date_oldest          Park
## 1    Crater Lake     Oregon       1902-05-22  1902-05-22    Crater Lake
## 2        Glacier    Montana       1910-05-11  1910-05-11        Glacier
## 3     Mesa Verde   Colorado       1906-06-29  1906-06-29     Mesa Verde
## 4  Mount Rainier Washington       1899-03-02  1899-03-02  Mount Rainier
## 5 Rocky Mountain   Colorado       1915-01-26  1915-01-26 Rocky Mountain
## 6        Sequoia California       1890-09-25  1890-09-25        Sequoia
```

```
#combining the created two dataframes by the names of the park in order
#to plot the average number of visitors
full_state_visit<-inner_join(clean_parks_abb, annual_report, by = c("Name" = "Park_name"))
```

```
#This plot visualizes the average number of visitors per year to the national parks
# in the corresponding states over the period of 10 years.
full_state_visit<-full_state_visit %>%
  dplyr::rename(state = State_abb)
plot_usmap(data = full_state_visit, values = "Average", labels = TRUE)
```